UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/676,552	09/30/2000	MICHAEL GINSBERG	1018.111US1	6912

22801 7590 10/31/2006

LEE & HAYES PLLC
421 W RIVERSIDE AVENUE SUITE 500
SPOKANE, WA 99201

EXAMINER	
BULLOCK JR, LEWIS ALEXANDER	
ART UNIT	PAPER NUMBER
2195	

DATE MAILED: 10/31/2006

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	09/676,552	GINSBERG, MICHAEL	
	Examiner	Art Unit	
	Lewis A. Bullock, Jr.	2195	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *02 August 2006*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-22* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-22* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

Claim Rejections - 35 USC § 101

1. 35 U.S.C. 101 reads as follows:

> Whoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefor, subject to the conditions and requirements of this title.

Claims 1-9 are rejected under 35 U.S.C. 101 because the claimed invention is directed to non-statutory subject matter. Claims 1-9 are directed to a non-functional data structure comprising of various linked list and an array. As proper under M.P.E.P. 2106, a non-functional data structure is non-statutory even if the data structure is stored on a machine readable medium. M.P.E. P. 2106 details:

If the "acts" of a claimed process manipulate only numbers, abstract concepts or ideas, or signals representing any of the foregoing, the acts are not being applied to appropriate subject matter. *Schrader*, 22 F.3d at 294-95, 30 USPQ2d at 1458-59. Thus, a process consisting solely of mathematical operations, i.e., converting one set of numbers into another set of numbers, does not manipulate appropriate subject matter and thus cannot constitute a statutory process.

In practical terms, claims define nonstatutory processes if they:
- consist solely of mathematical operations without some claimed practical application (i.e., executing a "mathematical algorithm"); or
- simply manipulate abstract ideas, e.g., a bid (*Schrader*, 22 F.3d at 293-94, 30 USPQ2d at 1458-59) or a bubble hierarchy (*Warmerdam*, 33 F.3d at 1360, 31 USPQ2d at 1759), without some claimed practical application.

A claim that requires one or more acts to be performed defines a process. However, not all processes are statutory under 35 U.S.C. 101. *Schrader*, 22 F.3d at 296, 30 USPQ2d at 1460. To be statutory, a claimed computer-related process must either: (A) result in a physical transformation outside the computer for which a practical application in the technological arts is either disclosed in the specification or would have been known to a skilled artisan (discussed in i) below), or (B) be limited to a practical application within the technological arts (discussed in ii) below). See *Diamond v. Diehr*, 450 U.S. at 183-84, 209 USPQ at 6 (quoting *Cochrane v. Deener*, 94 U.S. 780, 787-88 (1877)) ("A [statutory] process is a mode of treatment of certain materials to produce a given result. It is an act, or a series of acts, performed upon the subject-matter to be transformed and reduced to a different state or thing.... The process requires that certain things should be done with certain substances, and in a certain order; but the tools to be used in doing this may be of secondary consequence."). See also *Alappat*, 33 F.3d at 1543, 31 USPQ2d at 1556-57 (quoting *Diamond v. Diehr*, 450 U.S. at 192, 209 USPQ at 10). See also *id.* at 1569, 31 USPQ2d at 1578-79 (Newman, J., concurring) ("unpatentability of the principle does not defeat patentability of its practical applications") (citing *O 'Reilly v. Morse*, 56 U.S. (15 How.) at 114-19). If a physical transformation occurs outside the computer, a disclosure that permits a skilled artisan to practice the claimed invention, i.e., to put it to a practical use, is sufficient. On the other hand, it is necessary for the claimed invention taken as a whole to produce a practical application if there is only a transformation of signals or data inside a computer or if a process merely manipulates concepts or converts one set of numbers into another.

A process that merely manipulates an abstract idea or performs a purely mathematical algorithm is nonstatutory despite the fact that it might inherently have some usefulness. In *Sarkar*, 588 F.2d at 1335, 200 USPQ at 139, the court explained why this approach must be followed:

For such subject matter to be statutory, the claimed process must be limited to a practical application of the abstract idea or mathematical algorithm in the technological arts. See *Alappat*, 33 F.3d at 1543, 31 USPQ2d at 1556-57 (quoting *Diamond v. Diehr*, 450 U.S. at 192, 209 USPQ at 10). See also *Alappat* 33 F.3d at 1569, 31 USPQ2d at 1578-79 (Newman, J., concurring) ("unpatentability of the principle does not defeat patentability of its practical applications") (citing *O 'Reilly v. Morse*, 56 U.S. (15 How.) at 114-19). A claim is limited to a practical application when the method, as claimed, produces a concrete, tangible and useful result; i.e., the method recites a step or act of producing something that is concrete, tangible and useful. See *AT &T*, 172 F.3d at 1358, 50 USPQ2d at 1452. Likewise, a machine claim is statutory when the machine, as claimed, produces a concrete, tangible and useful result (as in *State Street*, 149 F.3d at 1373, 47 USPQ2d at 1601) and/or when a specific machine is being claimed (as in *Alappat*, 33 F.3d at 1544, 31 USPQ2d at 1557 (*> *en*< *banc*). For example, a computer process that simply calculates a mathematical algorithm that models noise is nonstatutory. However, a claimed process for digitally filtering noise employing the mathematical algorithm is statutory.

The claims refer to the data structure without any functionality and as detailed above all which would be non-statutory.

Claim Rejections - 35 USC § 112

2. The following is a quotation of the second paragraph of 35 U.S.C. 112:

> The specification shall conclude with one or more claims particularly pointing out and distinctly claiming the subject matter which the applicant regards as his invention.

3. Claims 1-10 and 14-18 are rejected under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention. Hence, Claims 1-10 and 14-18 are rejected under 35 U.S.C. 112, second paragraph, as being incomplete for omitting essential structural cooperative relationships of elements, such omission amounting to a gap between the necessary structural connections. See MPEP § 2172.01. The omitted

structural cooperative relationships are: the relationship between the various linked list and the array. Applicants invention is substantially defined in the specification and appeal brief as a new data structure wherein an array has entries each that relate to a particular rank range and point to the highest rank element of that range such that a linked list is formed from the highest rank element to subsequently higher rank elements of the rank range to form a linked list. The data structure further allows the highest rank element of each similar rank linked list to further linked to one another to form another linked list of highest rank entities (see figure 1 of the specification and pg. 4 of the appeal brief). All of the claims do not portray a resemblance of this data structure or its structural relationship. Claim 1 details a horizontally linked list and an array, however, the description of each structure details different entities. Therefore, the linked list and array have no relationship to each other, but can detail various structures for storing their separate range of entities. The same reasoning can be shown in relation to claim 10 and 14-18 wherein in removing a entity a determination can be made whether the entity is in various separate data structures for removal, i.e. a vertically linked list, a horizontally linked list, and an array. Claims 11-13 and 19-22 adequately portray the relationship between the elements of the structure. Applicant is referred to these claims as an example of how the elements should have the essential cooperative relationship as detailed in the appeal brief and the specification.

Claim Rejections - 35 USC § 103

4. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

5. Claims 1-22 are rejected under 35 U.S.C. 103(a) as being unpatentable over "Object Oriented Design for a Distributed Priority Queue" by Pen-Nan LEE et al. in view of "Shortest Path Algorithms" by GALLO et al.

As to claim 1, LEE teaches a data structure for efficiently ordering a plurality of entities (entities of a linked list), each entity having a rank (priority) within a plurality of ranks (priorities), the data structure comprising: a horizontally linked list (linked list of constituents that manage a linked list of their own wherein the constituents are arranged in the list order based on the priority range) linking at least a subset of the plurality of entities in at least a descending rank order direction (pg. 194, Class DPQ: Distributed Priority Queue Object, "...Each constituent object will maintain its part of the distributed state including: a range table containing the subset of ranges that each constituent object composing the distributed object is responsible for managing...a Next_id which indicates which constituent object contains the next lowest non-empty subrange of priority levels, an LPQ object for maintaining the elements which belong to the range of priorities which the constituent object is responsible for managing."; pg. 195, "Specifically, constituent objects will (among other things) need to do the following actions privately: Each constituent will have to pass the elements which it is asked to

insert into the distributed priority queue along to the constituent object which is responsible for maintaining the priority level for that element....Each of the constituents will need to communicate with each other in order to maintaining the proper ordering of elements by priority level."; see also pg. 195, DPQ Object Operations), each entity in the horizontally linked list having a unique rank as compared to the ranks of other entities in the horizontally linked list (each constituent having a priority level / range it is responsible for); and a table having a plurality of entries over which the plurality of ranks are distributed such that each table entry has a corresponding range of ranks, at least one table entry each indicating an entity of the plurality of entities having a greatest rank within the corresponding range of ranks for the table entry (via the table indicating the constituent that is responsible for handling a particular priority range and that constituent indicates the elements of the range) (see page 195). However, LEE does not allude to the data structure being stored on a medium and the table as an array having a plurality of array entries over which ranges of the ranks are distributed. LEE does teach program code that is used to implements and manipulates the priority queue objects and a table wherein the ranges of ranks are distributed (Range_table) (pg. 194). It is well known to one of ordinary skill in the art at the time of the invention that code can be stored on a medium and therefore would be obvious for one to do so in order to save, maintain, or distribute software. It is also inherent to the teachings of LEE that since ranges of priorities are handled by respective queues that their exists less table entries then possible priorities.

GALLO teaches a priority queue structure wherein a queue is issued to refer to the individual queues / linked lists (see page 22, 7.3 Buckets; pg. 6, fig. 1). It would be obvious by one of ordinary skill in the art at the time of the invention, that the LEE's table identification of the corresponding queues and their ranges of entities corresponds to GALLO's array wherein the array entries to the various linked lists are the table entries to the various ranged LPQ's. Therefore, it would be obvious to one of ordinary skill in the art to combine the teachings of LEE with the teachings of GALLO in order to manage access to data structures thereby improving searching performance (Lee, Introduction; GALLO, pg. 3-4, abstract and introduction).

As to claim 10, LEE teaches a method for removing a particular entity from a plurality of entities (via deleting the entity) as represented in a data structure for efficiently ordering the entities, the entity having respective ranks (priority) within a plurality of N ranks (priorities), the method comprising: in response to determining that the particular entity is present within a vertically linked list linking in at least one direction a corresponding subset of the plurality of entities having an identical rank, the corresponding subset including the particular entity, delinking the particular entity from the vertically linked list (via determining which constituent object contains the highest non-empty priority level and invoke that constituent objects delete operation wherein that object which contains the element to be deleted performs the deletion of the element from its LPQ object by invoking the LPQ object's delete operation) (pg. 194, "The operation DELETE..."; pg. 195, 4.2 Delete Operation); in response to determining

that the particular entity is present within a horizontally linked list (linked list of constituent objects) linking at least a subset of the plurality of entities in at least in a descending rank order direction, the subset including the particular entity (via determining which constituent object contains the highest non-empty priority level and invoke that constituent objects delete operation wherein that object which contains the element to be deleted performs the deletion of the element from its LPQ object by invoking the LPQ object's delete operation and if the element is the last element in the hed constituents LPQ, the constituent referenced in the Next_id of the head constituent should become the new head constituent) (pg. 194, "The operation DELETE..."; pg. 195, 4.2 Delete Operation), delinking the particular entity from the horizontally linked list (via setting the Next_id of the head constituent as the new head); and a table having a plurality of entries over which a plurality of rank ranges are distributed such that each table entry has a corresponding range of ranks, at least one table entry each indicating an entity of the plurality of entities having a greatest rank within the corresponding range of ranks for the table entry (via the table indicating the constituent that is responsible for handling a particular priority range and that constituent indicates the elements of the range) (see page 195). However, LEE does not allude to the data structure being stored on a medium and the table as an array having a plurality of array entries over which ranges of the ranks are distributed. LEE does teach program code that is used to implements and manipulates the priority queue objects and a table wherein the ranges of ranks are distributed (Range_table) (pg. 194). It is well known to one of ordinary skill in the art at the time of the invention that code can be stored on a medium and therefore

would be obvious for one to do so in order to save, maintain, or distribute software. It is also inherent to the teachings of LEE that since ranges of priorities are handled by respective queues that their exists less table entries then possible priorities. LEE also teaches that when removing an entity from the queue if the queue becomes empty, then the table entry is adjusted nil (pg. 195, "For a DELETE operation, if there is at least one element...If the DELETE operation causes the DPQ to become empty, then the head constitutuent which contains the last element simply changes its Head_id to NIL.")

GALLO teaches a priority queue structure wherein a queue is issued to refer to the individual queues / linked lists (see page 22, 7.3 Buckets; pg. 6, fig. 1). It would be obvious by one of ordinary skill in the art at the time of the invention, that the LEE's table identification of the corresponding queues and their ranges of entities corresponds to GALLO's array wherein the array entries to the various linked lists are the table entries to the various ranged LPQ's. Therefore, it would be obvious to one of ordinary skill in the art to combine the teachings of LEE with the teachings of GALLO in order to manage access to data structures thereby improving searching performance (Lee, Introduction; GALLO, pg. 3-4, abstract and introduction).

As to claim 19, LEE teaches a method for adding a new entity (via the insert operation) having a rank (priority) within a plurality of ranks (priorities) to a plurality of entities (queues) as represented in a data structure for efficiently ordering the entitites, the entities also having respective ranks within the plurality of N ranks, the method comprising: of a plurality of table entries of a table over which the plurality of ranks are

distributed such that a table entry has a corresponding range of ranks, determining the table entry having the corresponding range of ranks in which the rank of the new entity lies (see pg. 194, "The operation INSERT takes an element and a priority level for that element and determines the constituent object to which the element should be forwarded by consulting the range table. Once the destination constituent object is known, the element is forwarded to that object by invoking its INSERT operation..."; pg. 195, DPQ Object Operations, "The DPQ is divided into segments. Each segment is assigned to a different constituent in the system...Each range contains an upper and lower bound. Any element whose priority falls in between the upper and lower bound is placed in that range...Whenever a constituent object receives an insert request message from either its local processes or other processes in the network...In the latter case, the non-empty constituent must find its proper position with respect to other non-empty constituents in the DPQ."); adjusting the table entry having the corresponding range of ranks into which the rank of the new entity lies to point to the new entity in response to determining that the table entry currently points to null (pg. 195, "Initially each constituent will have the above range table and all id variables will be initialized to NIL except that the Head_ids of LDPQ2 through LPDQ5 are initialized to the id of LDPQ1....In the former case, the constituent just simple claims itself to be the head constituent of the DPQ."); adjusting the array entry having the corresponding range of ranks into which the rank of the new entity lies to point to the new entity in response to determining that the array entry currently points to an entity having a rank less than the rank of the new entity (via the Hookup Operation) (pg. 195-196, "A constituent issues a

HOOKUP request message when its status changes from empty to non-empty. The proper position of this constituent with respect to other existing non-empty constituents in the DPQ is determined by the priority level it holds. If its priority range is the highest among the existing non-empty constituents then it will become the new head constituent. In this case, it will be placed in from of the current head constituent..."); linking the new entity into a vertically linked list linking in at least one direction a corresponding subset of the plurality of entities having an identical rank, in response to determining that the rank of the new entity is equal to the rank of any other entity within the plurality of entities (via inserting the entity on the queue) (pg. 194, and 195); and otherwise, linking the new entity into a horizontally linked list linking at least a subset of the plurality of entities in at least a descending rank order direction, each entity in the horizontally linked list having a unique rank as compared to the ranks of the other entities in the horizontally linked list (via constituent not being empty and thereby linking the constituents in order) (pg. 195 – 196). However, LEE does not allude to the data structure being stored on a medium and the table as an array having a plurality of array entries over which ranges of the ranks are distributed. LEE does teach program code that is used to implements and manipulates the priority queue objects and a table wherein the ranges of ranks are distributed (Range_table) (pg. 194). It is well known to one of ordinary skill in the art at the time of the invention that code can be stored on a medium and therefore would be obvious for one to do so in order to save, maintain, or distribute software. It is also inherent to the teachings of LEE that since ranges of

priorities are handled by respective queues that their exists less table entries then possible priorities.

GALLO teaches a priority queue structure wherein a queue is issued to refer to the individual queues / linked lists (see page 22, 7.3 Buckets; pg. 6, fig. 1). It would be obvious by one of ordinary skill in the art at the time of the invention, that the LEE's table identification of the corresponding queues and their ranges of entities corresponds to GALLO's array wherein the array entries to the various linked lists are the table entries to the various ranged LPQ's. Therefore, it would be obvious to one of ordinary skill in the art to combine the teachings of LEE with the teachings of GALLO in order to manage access to data structures thereby improving searching performance (Lee, Introduction; GALLO, pg. 3-4, abstract and introduction).

As to claim 2, LEE teaches the data structure comprising at least one vertically linked list, each vertically linked list linking in at least one direction a corresponding subset of the plurality of entities having an identical rank (the LPQ of a constituent) (pg. 194, "Each constituent object will maintain...an LPQ object for maintaining the elements which belong to the range of priorities which the constituent object is responsible for managing.").

As to claim 3, LEE teaches each vertically linked list links the corresponding subset of the plurality of entities in a vertical direction (pg. 194, "Each constituent object

will maintain...an LPQ object for maintaining the elements which belong to the range of priorities which the constituent object is responsible for managing."). However, the cited combination does not allude to the object has having a second direction. Official Notice is taken in that it is well known in the art that double linked list are well known in the art and that the linked list queue of LEE is a well known double linked list that is capable of having two directions.

As to claim 4, LEE teaches the data structure further comprises a head pointer pointing to an entity having a greatest rank of the plurality of ranks of the plurality of entities (Head_id) (pg. 194).

As to claim 5, LEE teaches the horizontally linked list (list of constituents from head on down based on the next_id pointer) further links at least the subset of the plurality of entities in an ascending rank order direction (via the hookup operation) (pg. 195 – 196).

As to claim 6, LEE teaches the plurality of ranks are distributed over the plurality of array entries (pg. 195, fig. 1). It would be an obvious design choice that this distribution is equal.

As to claim 7, LEE teaches the entity having the greatest rank within the corresponding range of ranks for each of one or more of the at least one table entry is

one of a subset of the plurality of entities having the greatest rank within the corresponding range of ranks for the table entry (via indicating the highest range based on head_id wherein the head constituent has a plurality of elements) (pg. 195).

As to claim 8, LEE teaches at least one table entry of the plurality of table entries each points to null, corresponding to no entity within the plurality of entities having a rank within the corresponding range of ranks for the table entry (via initializing the head_ids to nil) (pg. 195, DPQ object Operations).

As to claim 9, LEE and GALLO when combined teach the array is a priority queue and that the rank of an entity is its priority. However, neither LEE nor GALLO detail that the entities are threads. Official Notice is taken in that it is well known in the art that threads are entities that are queued for serviced according to some priority structure and therefore would be obvious to one of ordinary skill in the art that threads are queued using the data structure of LEE and GALLO.

As to claim 11, LEE teaches the table entry has a corresponding range of ranks, and adjusting the table entry to indicate one of null and another one of the plurality of entities comprises, in response to determining that the particular entity was present within the vertically linked list, adjusting the table entry to indicate to a next entity within the vertically linked list (via the head pointer remaining pointing to the head constituent which would indicate a new first element) (pg. 195, "For a DELETE operation, if there is

at least one element in the DPQ, the first element of the highest non-empty priority level in the head constituent will be deleted from the constituent's LPQO.").

As to claim 12, LEE teaches adjusting the table entry to point to one of null and another one of the plurality of entries comprises, otherwise in response to determining that the particular entity was present within the horizontally linked list, and that the rank of the next entity within the horizontally linked list is within the corresponding range of ranks for the array entry, adjusting the table entry to indicate to the next entity within the horizontally linked list (pg.195, "When a delete operation removes the last element from the head constituent's LPQ , the constituent referenced in the Next_id of the head constituent should become the new head constituent.")

As to claim 13, LEE also teaches that when removing an entity from the queue if the queue becomes empty, then the table entry is adjusted nil (pg. 195, "For a DELETE operation, if there is at least one element...If the DELETE operation causes the DPQ to become empty, then the head constitutuent which contains the last element simply changes its Head_id to NIL.")

GALLO teaches a priority queue structure wherein a queue is issued to refer to the individual queues / linked lists (see page 22, 7.3 Buckets; pg. 6, fig. 1). It would be obvious by one of ordinary skill in the art at the time of the invention, that the LEE's table identification of the corresponding queues and their ranges of entities corresponds to GALLO's array wherein the array entries to the various linked lists are the table

entries to the various ranged LPQ's. Therefore, it would be obvious to one of ordinary skill in the art to combine the teachings of LEE with the teachings of GALLO in order to manage access to data structures thereby improving searching performance (Lee, Introduction; GALLO, pg. 3-4, abstract and introduction).

As to claim 14, LEE teaches in response to determining that a head pointer pointing to an entity having a greatest rank of the plurality of ranks of the plurality of entities points to a particular entity, adjusting the head pointer to point to another one of the plurality of entities (pg.195, "When a delete operation removes the last element from the head constituent's LPQ , the constituent referenced in the Next_id of the head constituent should become the new head constituent.").

As to claim 15, LEE teaches adjusting the head pointer to point to another one of the plurality of entities comprises, in response to determining that the particular entity was present within the vertically linked list, adjusting the head pointer to point to a next entity within the vertically linked list (via the head pointer remaining pointing to the head constituent which would indicate a new first element) (pg. 195, "For a DELETE operation, if there is at least one element in the DPQ, the first element of the highest non-empty priority level in the head constituent will be deleted from the constituent's LPQO.").

As to claim 16, LEE teaches adjusting the head pointer to point to another one of the plurality of entities comprises, otherwise in response to determining that the particular entity was present within the horizontally linked list, adjusting the head pointer to point to a next entity within the horizontally linked list (pg.195, "When a delete operation removes the last element from the head constituent's LPQ , the constituent referenced in the Next_id of the head constituent should become the new head constituent.").

As to claim 17, LEE and GALLO when combined teach the array is a priority queue and that the rank of an entity is its priority. However, neither LEE nor GALLO detail that the entities are threads. Official Notice is taken in that it is well known in the art that threads are entities that are queued for serviced according to some priority structure and therefore would be obvious to one of ordinary skill in the art that threads are queued using the data structure of LEE and GALLO.

As to claim 18, refer to claim 10 for rejection.

As to claim 20, LEE teaches adjusting a head pointer to an entity having the greatest rank of a plurality of ranks of the plurality of entities to point to the new entity in response to determining that the rank of the new entity is greater than the rank of the entity of the current head pointer (via performing a HOOKUP Operation and determining

the proper position of the linked list / constituents based on the priority range) (pg. 195-196).

As to claim 21, LEE and GALLO when combined teach the array is a priority queue and that the rank of an entity is its priority. However, neither LEE nor GALLO detail that the entities are threads. Official Notice is taken in that it is well known in the art that threads are entities that are queued for serviced according to some priority structure and therefore would be obvious to one of ordinary skill in the art that threads are queued using the data structure of LEE and GALLO.

As to claim 22, refer to claim 19 for rejection.

Response to Arguments

6. Applicant's arguments filed August 2, 2006 have been fully considered but they are not persuasive. Regarding the 101 issues, Applicant is alluding to claims 1-22 as functional material. The examiner agrees that claims 10-22 are functional material because these claims detail actual steps for removing or adding an entity into the data structure to effect the ordering of the entities, thereby giving patentable weight to the preamble. However, claims 1-9 only relates to the arrangement of the data structure. As outlined in the section regarding M.P.E.P. 2106, a mere arrangement of data, herein the mere arrangement of entities is non-functional descriptive material. The intended use of the claims, "for efficiently ordering a plurality of entities" is not realized because

the entities are already arranged. There are no functional steps for adding entities or removing entities to realize the ordering of the plurality of entities as in the other claims. The claims are only related to the arrangement of the data structure. Therefore, these claims are non-functional and still non-statutory.

Regarding the 112 2nd paragraph rejections, Applicant argues that neither Applicants specification nor any statements on record describe or define any subject matter as being essential and further that the linked list and the array are related based on the "at least one of the N array entries [is] pointing to an entity of the plurality of entities of the linked list. The examiner disagrees. Regarding the claims as disclosed, the cited passage details in the preamble **a plurality of entities wherein the entities have a respective ranks** within a plurality of ranks. The cited data structure comprises a horizontally linked list linking **at least a subset of the plurality of entities** wherein the entities in the horizontally linked list have respective ranks as compared to other entities in the linked list. The data structure also has an array having **a plurality of fewer than N array entries pointing to an entity of the plurality of entities having a greatest rank within the range of ranks corresponding to the at least one array entry**. In order to interpret the cited data structure, the examiner must make a determination of whether the subset of the entities relating to the linked are either (1) the same referenced entities in the array or (2) the remaining subset of the plurality of entities having no relation to the linked list subset. Broadly interpreting the claim, the only requirement is that all entities have a rank and that the linked list comprises **at least a** subset these entities. There is no requirement in the claim that the entities

pointed to by the array entries are the same entities of the linked list. The argued and italic limitations allude to the array entries pointing to **an entity** of the plurality of entities. The preamble established that there exists a plurality of entities. Equating the array entry entity to the linked list entity would be reading this limitation into the claim language, which is improper under M.P.E.P. 2111. Therefore, in the broadest possible interpretation of the claims consistent with the specification, the linked list has no relationship to the array. Regarding the assertion that the elements being essential is not previously made, the examiner points to the appeal brief. The summary of the invention of the appeal brief refers to figure 1 and illustrates that the array entries point to elements of the linked list. As outlined above, this is not claimed and clearly is essential because it's the foundation of the data structure Applicant is filing a patent on. Entities are ordered in the data structure by manipulating this data structure through not only array entries by also linked list that are referenced by the array entries. The cited claims maintain these sub-structures separate such that each functions with its own subset of entities and have no relation to one another. This is contrary to the appeal brief and the specification. Regarding claim 10, the examiner reiterates that the determining operation in each of the cited steps, makes a judgment of whether the entity to be removed I present within a vertically linked list, a horizontally linked list, or an array element. If the linked lists and array elements had a relationship and each made up the overall data structure, the element would be in all structures, e.g. the linked lists (vertical and horizontal) and the array. By separately determining whether the element is in either the vertically linked list, the horizontally linked list or the array,

the claims can be interpreted that each is separate and have no relation to one another which is adverse from the teachings of the appeal brief and the specification. Figure 1 establishes that the array entries point to elements in the horizontally linked list which each is the head element to entities in the vertical linked list. The claim does not allude to this.

7. Applicant's arguments with respect to the prior art rejections of claims 1-22 have been considered but are moot in view of the new ground(s) of rejection.

Conclusion

8. Applicant's amendment necessitated the new ground(s) of rejection presented in this Office action. Accordingly, **THIS ACTION IS MADE FINAL**. See MPEP § 706.07(a). Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the date of this final action.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Lewis A. Bullock, Jr. whose telephone number is (571) 272-3759. The examiner can normally be reached on Monday-Friday, 8:30 a.m. - 5:00 p.m..

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Meng An can be reached on (571) 272-3756. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

October 28, 2006

LEWIS A. BULLOCK, JR.
PRIMARY EXAMINER